Performance Analysis of Block Markov Superposition Transmission of Short Codes

Kechao Huang, *Student Member, IEEE,* and Xiao Ma, *Member, IEEE*

***Abstract*—In this paper, we consider the asymptotic and finite-length performance of block Markov superposition transmission (BMST) of short codes, which can be viewed as a new class of spatially coupled (SC) codes with the generator matrices of short codes (referred to as *basic codes*) coupled. A modified extrinsic information transfer (EXIT) chart analysis that takes into account the relation between mutual information (MI) and bit-error-rate (BER) is presented to study the convergence behavior of BMST codes. Using the modified EXIT chart analysis, we investigate the impact of various parameters on BMST code performance, thereby providing theoretical guidance for designing and implementing practical BMST codes suitable for sliding window decoding. Then, we present a performance comparison of BMST codes and SC low-density parity-check (SC-LDPC) codes on the basis of equal decoding latency. Also presented is a comparison of computational complexity. Simulation results show that, under the equal decoding latency constraint, BMST codes using the repetition code as the basic code can outperform $(3,6)$-regular SC-LDPC codes in the waterfall region but have a higher computational complexity.**

***Index Terms*—Block Markov superposition transmission (BMST), capacity-approaching codes, extrinsic information transfer (EXIT) chart analysis, sliding window decoding, spatial coupling.**

I. Introduction

Low-density parity-check (LDPC) block codes (LDPC-BCs) [1], combined with iterative belief propagation (BP) decoding, are a class of capacity-approaching codes with decoding complexity that increases only linearly with block length [2]. A practical approach to improving the performance of LDPC-BCs is coupling together a series of L disjoint graphs that specify the parity-check matrix of an LDPC-BC into a single coupled chain, thereby producing a spatially coupled LDPC (SC-LDPC) code. It has been shown in [3–6] that SC-LDPC code ensembles exhibit a phenomenon called "threshold saturation", which allows them to achieve the maximum *a posteriori* (MAP) thresholds of their underlying LDPC-BC ensembles on memoryless binary-input symmetric-output channels under BP decoding, and thus to achieve capacity by increasing the density of the parity-check matrix. Due to their excellent performance, SC-LDPC codes have recently received a great deal of attention in the literature (see, e.g., [7–15] and the references therein).

The concept of spatial coupling is not limited to LDPC codes. Block Markov superposition transmission (BMST) of

This work was partially supported by the 973 Program (No. 2012CB316100) and the China NSF (No. 91438101 and No. 61172082).

The authors are with the Department of Electronics and Communication Engineering, Sun Yat-sen University, Guangzhou, GD 510006, China (e-mail: hkech@mail2.sysu.edu.cn; maxiao@mail.sysu.edu.cn).

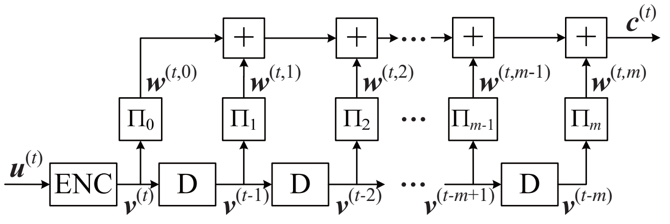


Fig. 1. Encoder of a BMST code with encoding memory m, where the information sequence $\boldsymbol{u}^{(t)}$ at time t is encoded into the sub-codeword $\boldsymbol{c}^{(t)}$ for transmission.

short codes [16, 17], for example, is equivalent to spatial coupling of the subgraphs that specify the generator matrices of the short codes. From this perspective, BMST codes are similar to braided block/convolutional codes [18–20], staircase codes [21], and SC turbo codes [22]. An encoder of a BMST code with encoding memory m is shown in Fig. 1, where a BMST code can also be viewed as a serially concatenated code with a structure similar to repeat-accumulate-like codes [23–25]. The outer code is a short code, referred to as the *basic code* (not limited to repetition codes), that introduces redundancy, while the inner code is a rate-one block-oriented feed-forward convolutional code (instead of a bit-oriented accumulator) that introduces memory between transmissions. Hence, BMST codes typically have very simple encoding algorithms. To decode BMST codes, a sliding window decoding algorithm with a tunable decoding delay can be used, as with SC-LDPC codes [26]. The construction of BMST codes is flexible [27, 28], in the sense that it applies to all code rates of interest in the interval (0,1). Further, BMST codes have near-capacity performance (observed by simulation) in the waterfall region of the bit-error-rate (BER) cruve and an error floor (predicted by analysis) that can be controlled by the encoding memory.

On an additive white Gaussian noise channel (AWGNC), the well-known extrinsic information transfer (EXIT) chart analysis [29] can be used to obtain the iterative BP decoding threshold of LDPC-BC ensembles. In [30], a novel EXIT chart analysis was used to evaluate the performance of protograph-based LDPC-BC ensembles, and a similar analysis was used to find the thresholds of q-ary SC-LDPC codes with sliding window decoding in [31]. Unlike LDPC codes, the asymptotic BER of BMST codes with window decoding cannot be better than a corresponding genie-aided lower bound [16]. Thus, conventional EXIT chart analysis cannot be applied directly to BMST codes. In this paper, we propose a modified EXIT chart analysis, that takes into account the relation between mutual information (MI) and BER, to study the convergence behavior of BMST codes and to predict the performance in the waterfall region of the BER curve. Simulation results confirm that the modified EXIT chart analysis of BMST codes is supported by their finite-length performance behavior. We also investigate the relationship between the basic code structure, the decoding delay, and the decoding performance of BMST codes when the decoding latency is fixed. Finally, we present a computational complexity comparison of BMST codes and SC-LDPC codes on the basis of equal decoding latency.

The rest of the paper is structured as follows. In Section II, we give a brief review of BMST codes. In Section III, we discuss the relation between BMST codes and protograph-based SC-LDPC codes. In Section IV, we propose a modified EXIT chart analysis of BMST codes. In Section V, we investigate the impact of various parameters on BMST code performance. Then, in Section VI, we present a performance comparison of BMST codes and SC-LDPC codes on the basis of equal decoding latency. A computational complexity comparison of BMST codes and SC-LDPC codes is also given in Section VI. Finally, some concluding remarks are given in Section VII.

II. Review of BMST Codes

A. Encoding of BMST Codes

Consider a BMST code using a rate $R = k/n$ binary basic code $\mathscr{C}[n, k]$ of length n and dimension k. Let $\boldsymbol{u} = (\boldsymbol{u}^{(0)}, \boldsymbol{u}^{(1)}, \cdots, \boldsymbol{u}^{(L-1)})$ be L blocks of data to be transmitted, where $\boldsymbol{u}^{(t)} \in \mathbb{F}_2^k$. Here, L is called the *coupling length*. The encoding algorithm of a BMST code with encoding memory (*coupling width*) m is described as follows (see Fig. 1), where $\boldsymbol{\Pi}_i$ $(0 \le i \le m)$ are $m+1$ interleavers of size n.

***Algorithm** 1:* Encoding of BMST Codes

```
• Initialization: For $t < 0$, set $\boldsymbol{v}^{(t)} = \mathbf{0} \in \mathbb{F}_2^n$.
• Loop: For $t = 0, 1, \cdots, L-1$,
    1) Encode $\boldsymbol{u}^{(t)}$ into $\boldsymbol{v}^{(t)} \in \mathbb{F}_2^n$ using the encoding algorithm of the basic code $\mathscr{C}$;
    2) For $0 \le i \le m$, interleave $\boldsymbol{v}^{(t-i)}$ using the $i$-th interleaver $\boldsymbol{\Pi}_i$ into $\boldsymbol{w}^{(t,i)}$;
    3) Compute $\boldsymbol{c}^{(t)} = \sum_{0 \le i \le m} \boldsymbol{w}^{(t,i)}$, which is taken as the $t$-th block of transmission.
• Termination: For $t = L, L+1, \cdots, L+m-1$, set $\boldsymbol{u}^{(t)} = \mathbf{0} \in \mathbb{F}_2^k$ and compute $\boldsymbol{c}^{(t)}$ following Loop.
```

Remark: To force the encoder of BMST codes to the zero state at the end of the encoding process, a tail consisting of m blocks of the k-dimensional all-zero vector is added. This is different from SC-LDPC code encoders, where the tail is usually non-zero and depends on the encoded information bits (see Section IV of [32]). As a result, the termination procedure for BMST codes is much simpler than for SC-LDPC codes.

The rate of the BMST code is

$$R_{\text{BMST}} = \frac{Lk}{(L+m)n} = \frac{L}{L+m}R, \tag{1}$$

which is slightly less than the rate $R = k/n$ of the basic code. However, similar to SC-LDPC codes, this rate loss becomes vanishingly small as $L \to \infty$.

Though any code (linear or nonlinear) with a fast encoding algorithm and an efficient soft-in soft-out (SISO) decoding algorithm can be taken as the basic code, we focus in this paper on the use of the B-fold Cartesian product of a repetition (R) code (denoted by R $[N, 1]$) or a single parity-check (SPC) code (denoted by SPC $[N, N-1]$) as the basic code, resulting in a BMST-R code (denoted by BMST-R $[N, 1]$) or a BMST-SPC code (denoted by BMST-SPC $[N, N-1]$), respectively.[1] Note that the overall code length of the basic code in this case is $n = BN$ and the overall dimension is $k = B$ or $B(N-1)$.

B. Sliding Window Decoding of BMST Codes

BMST codes can be represented by a Forney-style factor graph, also known as a normal graph [33], where edges represent variables and vertices (nodes) represent constraints. All edges connected to a node must satisfy the specific constraint of the node. A full-edge connects to two nodes, while a half-edge connects to only one node. A half-edge is also connected to a special symbol, called a "dongle", that denotes coupling to other parts of the transmission system (say, the channel or the information source) [33]. There are four types of nodes in the normal graph of BMST codes.

- **Node** $\boxed{+}$: All edges (variables) connected to node $\boxed{+}$ must sum to the all-zero vector. The message updating rule at node $\boxed{+}$ is similar to that of a check node in the factor graph of a binary LDPC code. The only difference is that the messages on the half-edges are obtained from the channel observations.
- **Node** $\boxed{\Pi_i}$: The node $\boxed{\Pi_i}$ represents the i-th interleaver, which interleaves or de-interleaves the input messages.
- **Node** $\boxed{=}$: All edges (variables) connected to node $\boxed{=}$ must take the same (binary) values. The message updating rule at node $\boxed{=}$ is the same as that of a variable node in the factor graph of a binary LDPC code.
- **Node** $\boxed{\text{G}}$: All edges (variables) connected to node $\boxed{\text{G}}$ must satisfy the constraint specified by the basic code $\mathscr{C}$. The message updating rule at node $\boxed{\text{G}}$ can be derived accordingly, where the messages on the half-edges are associated with the information source.

The normal graph of a BMST code can be divided into *layers*, where each layer typically consists of a node of type $\boxed{\text{G}}$, a node of type $\boxed{=}$, m nodes of type $\boxed{\Pi}$, and a node of type $\boxed{+}$ (see Fig. 2). The result is a high-level normal graph, where each edge represents a sequence of random variables. Looking into the details, we can see that, at each layer, there are n nodes $\boxed{=}$ of degree $m+2$, n nodes $\boxed{+}$ of degree $m+2$ (including half edges), and B nodes $\boxed{\text{G}_0}$ corresponding to the short code (R $[N, 1]$ or SPC $[N, N-1]$ in this paper).

Similar to SC-LDPC codes, an iterative sliding window decoding algorithm with decoding delay d working over a subgraph consisting of $d+1$ consecutive layers can be implemented for BMST codes. An example of a window decoder with decoding delay $d = 2$ operating on the normal

[1]Using codes constructed by time-sharing between the R code and the SPC code as the basic code, one can construct BMST-RSPC codes for a wide range of code rates. For more details, see [28].

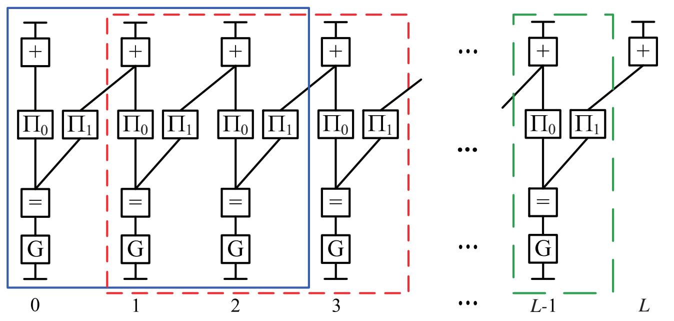


Fig. 2. Example of a window decoder with decoding delay $d = 2$ operating on the normal graph of a BMST code with $m = 1$ at times $t = 0$ (solid blue) and $t = 1$ (dotted red). For each window position/time instant, the first (left-most) decoding layer is called the target layer.

graph of a BMST code with $m = 1$ is shown in Fig. 2. For each window position, the forward-backward decoding algorithm is implemented for updating the messages layer-by-layer within the decoding window.[2] Decoding proceeds until a fixed number of iterations has been performed or some given stopping criterion is satisfied, in which case the window shifts to the right by one layer and the symbols corresponding to the layer shifted out of the window are decoded. The first layer in any window is called the *target layer*.

C. Genie-Aided Lower Bound on BER

Let $p_b = f_{\rm BMST}(\gamma_b)$ represent the performance of a BMST code with encoding memory (coupling width) m and coupling length L, where p_b is the BER and $\gamma_b \triangleq E_b/N_0$ represents the received bit signal-to-noise ratio (SNR) on an AWGNC in dB, and let $p_b = f_{\rm Basic}(\gamma_b)$ represent the performance of the basic code. By assuming a genie-aided decoder, we can obtain a lower bound on the performance of BMST codes given by (see [16])

$$f_{\rm BMST}(\gamma_b) \geq f_{\rm Basic}(\gamma_b + 10\log_{10}(m+1) - 10\log_{10}(1+m/L)), \tag{2}$$

where the term $10\log_{10}(m+1)$ depends on the encoding memory m and the term $10\log_{10}(1+ m/L)$ is due to the rate loss. In other words, a maximum coding gain over the basic code of $10\log_{10}(m+1)$ dB in the low BER (high SNR) region is achieved for large L. Intuitively, this bound can be understood by assuming that a codeword in the basic code is transmitted $m+1$ times without interference from other layers.

D. Design of Capacity Approaching BMST Codes

Aided by the genie-aided lower bound, we can construct good codes at a target BER with any given code rate of interest by determining as follows the required encoding memory m.

1) Take a code with the given rate as the basic code. To approach channel capacity, we set the code length $n \geq 10000$;
2) From the performance curve $f_{\rm Basic}(\gamma_b)$ of the basic code, find the required $E_b/N_0 = \gamma_{\rm target}$ to achieve the target BER;

[2]For more details on the decoding algorithm of BMST codes, we refer the reader to Section III of [16].

TABLE I
ENCODING MEMORIES FOR BMST CODES REQUIRED TO APPROACH THE CORRESPONDING SHANNON LIMITS AT GIVEN TARGET BERs

Encoding memory m	Target BER			
	10^{-3}	10^{-4}	10^{-5}	10^{-6}
BMST-R $[2,1]$	4	6	8	10
BMST-R $[4,1]$	5	8	10	13
BMST-R $[8,1]$	6	9	11	14
BMST-SPC $[4,3]$	2	3	4	5

3) Find the Shannon limit for the code rate, denoted by $\gamma_{\rm lim}$;
4) Determine the encoding memory m by

$$m = \left\lceil 10^{\frac{\gamma_{\rm target}-\gamma_{\rm lim}}{10}} - 1 \right\rceil, \tag{3}$$

where $\lceil x \rceil$ represents the smallest integer greater than or equal to x.

The above procedure requires no optimization and hence can be easily implemented given that the performance curve $f_{\rm Basic}(\gamma_b)$ is available, as is the usual case for short codes.[3] Its effectiveness has been confirmed by construction examples in [16, 17, 27, 28]. The encoding memories for some BMST codes required to approach the corresponding Shannon limits at given target BERs are shown in Table I. As expected, the lower the target BER is, the larger the required encoding memory m is.

III. BMST Codes as a Class of SC Codes

In this section, we show that BMST codes can be viewed as a class of SC codes, using an algebraic description as well as a graphical representation, and we compare the structure of BMST codes to SC-LDPC codes.

A. Matrix Representation

To describe an SC-LDPC code ensemble with coupling width (*syndrome former memory*) m and coupling length L, we start with an $(L+m)(N-K) \times LN$ matrix

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_0 & & & \\ \boldsymbol{B}_1 & \boldsymbol{B}_0 & & \\ \vdots & \boldsymbol{B}_1 & \ddots & \\ \boldsymbol{B}_m & \vdots & \ddots & \boldsymbol{B}_0 \\ & \boldsymbol{B}_m & \ddots & \boldsymbol{B}_1 \\ & & \ddots & \vdots \\ & & & \boldsymbol{B}_m \end{bmatrix}, \tag{4}$$

where all of the $m+1$ component submatrices $\boldsymbol{B}_0, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_m$ have non-negative integer entries and size $(N-K) \times N$. To construct an SC-LDPC code with good performance, we can replace each non-zero entry $b \neq 0$ in $\boldsymbol{B}$

[3]The basic code considered in this paper is a Cartesian product of a short code, where each codeword is indeed a cascade of B separate and independent codewords from the short code. Thus, the performance of the basic code can easily be obtained, which is the same as that of the involved short code.

with a sum of b nonoverlapping randomly selected $M \times M$ permutation matrices and each zero entry in $\boldsymbol{B}$ with the $M \times M$ all-zero matrix, where b is typically a small integer and M is typically a large integer. The resulting SC-LDPC parity-check matrix $\boldsymbol{H}_{\mathrm{SC}}$ of size $(L+m)(N-K)M \times LNM$ is given by

$$\boldsymbol{H}_{\mathrm{SC}} = \begin{bmatrix} \boldsymbol{H}_0(0) & & & \\ \boldsymbol{H}_1(1) & \boldsymbol{H}_0(1) & & \\ \vdots & \boldsymbol{H}_1(2) & \ddots & \\ \boldsymbol{H}_m(m) & \vdots & \ddots & \boldsymbol{H}_0(L-1) \\ & \boldsymbol{H}_m(m+1) & \ddots & \boldsymbol{H}_1(L) \\ & & \ddots & \vdots \\ & & & \boldsymbol{H}_m(L+m-1) \end{bmatrix}, \quad (5)$$

where the blank spaces in $\boldsymbol{H}_{\mathrm{SC}}$ correspond to zeros and the submatrices $\boldsymbol{H}_i(t+i)$ have size $(N-K)M \times NM$, for $0 \le i \le m$ and $0 \le t \le L-1$.

In contrast to SC-LDPC codes, it is convenient to describe BMST codes using generator matrices. Let $\boldsymbol{G}_0$ be the generator matrix of a short code with dimension K and length N. To describe a BMST code ensemble with coupling width (*encoding memory*) m and coupling length L, we start with the $L \times (L+m)$ matrix

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & \cdots & 1 & & & & \\ & 1 & 1 & \cdots & 1 & & & \\ & & \ddots & \ddots & \ddots & \ddots & & \\ & & & 1 & 1 & \cdots & 1 & \\ & & & & 1 & 1 & \cdots & 1 \end{bmatrix}, \quad (6)$$

which has constant weight $m+1$ in each row. This matrix $\boldsymbol{A}$ plays a similar role for constructing BMST codes as the matrix $\boldsymbol{B}$ does for constructing SC-LDPC codes. To construct a BMST code with good performance, each nonzero entry $A_{j,j+i}$ ($0 \le j \le L-1$ and $0 \le i \le m$) in $\boldsymbol{A}$ is replaced with a matrix $\boldsymbol{G\Pi}_i$, where

$$\boldsymbol{G} = \mathrm{diag}\{\underbrace{\boldsymbol{G}_0, \cdots, \boldsymbol{G}_0}_{B}\} \quad (7)$$

is the generator matrix of the B-fold Cartesian product of the short code, the $\boldsymbol{\Pi}_i$ ($0 \le i \le m$) are $m+1$ randomly selected $NB \times NB$ permutation matrices, and the Cartesian product order B is typically large. The resulting BMST code has length $(L+m)NB$ and dimension LKB, and the generator matrix $\boldsymbol{G}_{\mathrm{BMST}}$ is given by

$$\boldsymbol{G}_{\mathrm{BMST}} = \begin{bmatrix} \boldsymbol{G\Pi}_0 & \boldsymbol{G\Pi}_1 & \cdots & \boldsymbol{G\Pi}_m & & & \\ & \boldsymbol{G\Pi}_0 & \boldsymbol{G\Pi}_1 & \cdots & \boldsymbol{G\Pi}_m & & \\ & & \ddots & \ddots & \ddots & \ddots & \\ & & & \boldsymbol{G\Pi}_0 & \boldsymbol{G\Pi}_1 & \cdots & \boldsymbol{G\Pi}_m \end{bmatrix}. \quad (8)$$

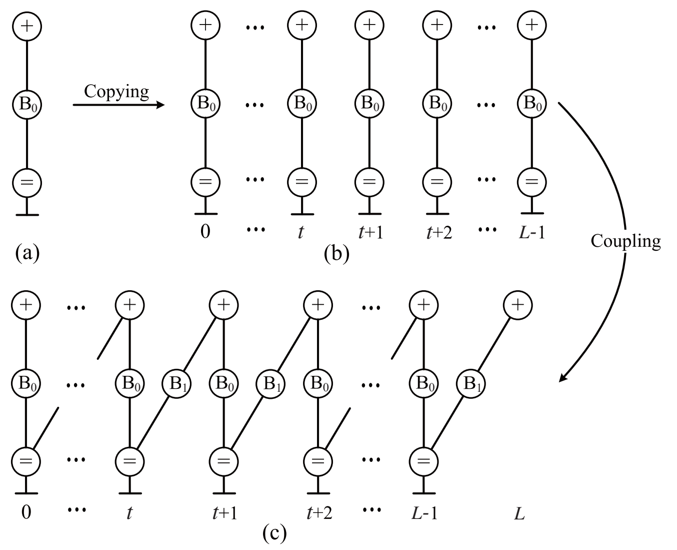


Fig. 3. (a) A protograph corresponding to the submatrix $\boldsymbol{B}_0$, (b) L uncoupled protographs, each of which corresponds to the submatrix $\boldsymbol{B}_0$, and (c) a protograph corresponding to an SC-LDPC code ensemble with coupling length L and coupling width $m=1$.

B. Graphical Representation

SC-LDPC code ensembles are often described in terms of a protograph, where an *edge-spreading* operation is applied to couple a sequence of disjoint block code protographs into a single chain [6]. Usually, no extra edges are introduced during the coupling process. In this paper, we describe the coupling process from a new perspective, where extra edges are allowed to be added. We believe that this new treatment is more general. For example, SC turbo codes [22] are obtained by adding edges to connect each turbo code graph to one or more nearby graphs in the chain. Based on this perspective, we can redescribe SC-LDPC codes as follows.

We start with a protograph for the submatrix $\boldsymbol{B}_0 = [B_{i,j}]$, which has N variable nodes and $N-K$ check nodes, where the i-th check node is connected to the j-th variable node by $B_{i,j}$ edges. A short-hand protograph corresponding to $\boldsymbol{B}_0$ is shown in Fig. 3(a), where the node $\ominus$ represents N variable nodes, the node $\oplus$ represents $N-K$ check nodes, and the edge $\textcircled{\scriptsize B_0}$ represents a collection of $\sum B_{i,j}$ edges. To distinguish, the edge $\textcircled{\scriptsize B_0}$ is referred to as a *super-edge* of type $\boldsymbol{B}_0$, while the conventional edge in the full protograph is referred to as a *simple edge*. The short-hand protograph is then replicated L times, as shown in Fig. 3(b), meaning that the sequence of transmitted codewords satisfy independently the constraint $\boldsymbol{B}_0$. The L disjoint graphs are then coupled by adding a *super-edge* of type $\boldsymbol{B}_i$ to bridge the variable node $\ominus$ at time t and the check node $\oplus$ at time $t+i$, for $0 \le t \le L-1$ and $1 \le i \le m$, resulting in a single coupled chain corresponding to an SC-LDPC code ensemble with coupling length L and coupling memory m. An example of an SC-LDPC code ensemble with coupling memory $m=1$ is shown in Fig. 3(c). When lifting, each *simple edge* (not *super-edge*) is replaced by a bundle of M edges (permutation within the bundle is assumed), resulting in an SC-LDPC code with length LNM.

Similarly, BMST codes start with a protograph for the generator matrix $\boldsymbol{G}_0 = [G_{i,j}]$, which has K $\boxed{=}$ nodes

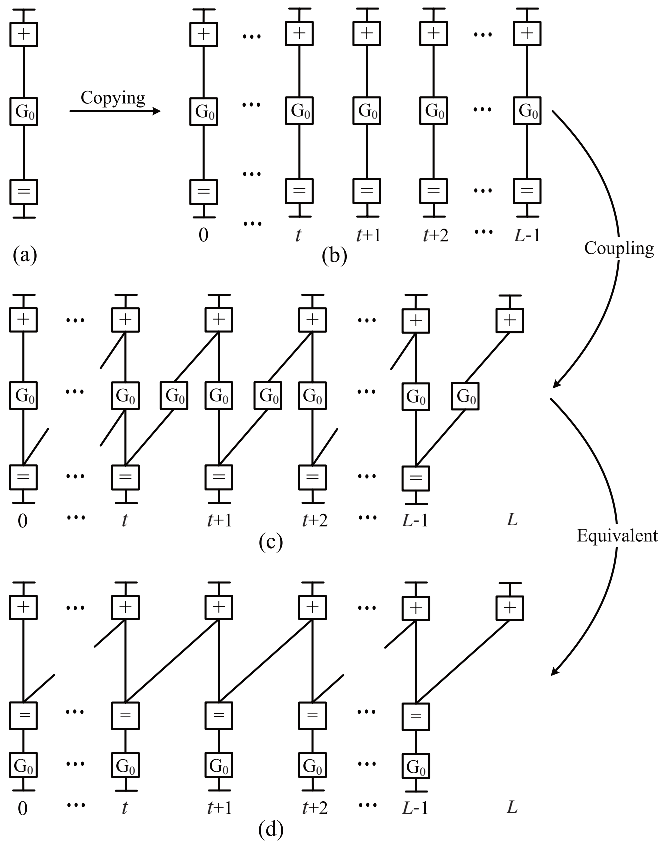


Fig. 4. (a) A protograph representing the basic code with generator matrix $\boldsymbol{G}_0$, (b) L uncoupled basic protographs, each of which corresponds to the generator matrix $\boldsymbol{G}_0$, (c) a protograph corresponding to a BMST code ensemble with coupling length L and coupling width $m = 1$, and (d) an equivalent protograph corresponding to the same BMST code ensemble with coupling length L and coupling width $m = 1$.

and N $\boxed{+}$ nodes, where the i-th $\boxed{=}$ node is connected to the j-th $\boxed{+}$ node if and only if $G_{i,j} = 1$. A short-hand protograph corresponding to $\boldsymbol{G}_0$ is shown in Fig. 4(a), where $\boxed{\mathrm{G}_0}$ represents a *super-edge* of type $\boldsymbol{G}_0$. The protograph is then replicated L times, as shown in Fig. 4(b), which can be considered as transmitting a sequence of codewords from the basic code corresponding to the generator matrix $\boldsymbol{G}_0$ independently at time instants $t = 0, 1, \cdots, L-1$. The L disjoint graphs are coupled by adding a *super-edge* of type $\boldsymbol{G}_0$ to bridge the $\boxed{=}$ node at time t and the $\boxed{+}$ node at time $t+i$, for $1 \leq i \leq m$, resulting in a single coupled chain corresponding to a BMST code ensemble with coupling length L and coupling memory m. An example of a BMST code ensemble with coupling memory $m = 1$ is shown in Fig. 4(c), whose equivalent form is shown in Fig. 4(d). When lifting, the *super-edge* of type $\boldsymbol{G}_0$ bridging the $\boxed{=}$ node at time t and the $\boxed{+}$ node at time $t+i$, for $0 \leq t \leq L-1$ and $0 \leq i \leq m$, is replaced by a *super-edge* of type $\boldsymbol{G}\boldsymbol{\Pi}_i$, resulting in a BMST code with length $(L+m)NB$.

C. Similarities and Differences

From the previous two subsections, we see that both SC-LDPC codes and BMST codes can be derived from a small matrix by replacing the entries with properly-defined sub-matrices. We also see that the generator matrix $\boldsymbol{G}_{\mathrm{BMST}}$ of BMST codes is similar in form to the parity-check matrix $\boldsymbol{H}_{\mathrm{SC}}$ of SC-LDPC codes. SC-LDPC codes introduce memory by spatially coupling the basic parity-check matrices $\boldsymbol{B}_0$, while BMST codes introduce memory by spatially coupling the basic generator matrices $\boldsymbol{G}_0$. Further, we see from Fig. 3 and Fig. 4 that during the construction of both SC-LDPC codes and BMST codes, the memory is introduced by coupling the disjoint graphs together in a single chain, which is the fundamental idea of spatial coupling. Thus, BMST codes can be viewed as a class of SC codes.

IV. EXIT Chart Analysis of BMST Codes

Given the basic code with generator matrix $\boldsymbol{G}_0$, we can construct a sequence of BMST codes by choosing the Cartesian product order $B = 1, 2, \cdots$. Now assume that the interleavers are chosen uniformly and at random for each transmission. Then we have a sequence of code ensembles. The aim of EXIT chart analysis is to predicte the performance behavior of the BMST codes as $B \to \infty$. In this section, we first discuss the issue that prevents the use of conventional EXIT chart analysis for BMST codes, and then we provide a modified EXIT chart analysis to study the convergence behavior of BMST codes with window decoding.

We consider binary phase-shift keying (BPSK) modulation over the binary-input AWGNC. To describe density evolution, it is convenient to assume that the all-zero codeword is transmitted and to represent the messages as log-likelihood ratios (LLRs). The threshold of protograph-based LDPC codes can be obtained based on a protograph-based EXIT chart analysis [30, 31] by determining the minimum value of the SNR E_b/N_0 such that the MI between the *a posteriori* message at a variable node and an associated codeword bit (referred to as the *a posteriori* MI for short) goes to 1 as the number of iterations increases, i.e., the BER at the variable nodes tends to zero as the number of iterations tends to infinity. At a first glance, a similar iterative sliding window decoding EXIT chart analysis algorithm can be implemented over the normal graph (see Fig. 4(d)) of the BMST code ensemble to study the convergence behavior of BMST codes. However, as shown in (2), the high SNR performance of BMST codes with window decoding cannot be better than the corresponding genie-aided lower bound, which means that the *a posteriori* MI of BMST codes cannot reach 1 as the number of iterations tends to infinity. Thus, the conventional EXIT chart analysis cannot be applied directly to BMST codes. Fortunately, this can be amended by taking into account the relation between MI and BER [29]. Specifically, we need the convergence check at node $\boxed{\mathrm{G}_0}$, as described below in Algorithm 2. For convenience, the MI between the *a priori* input and the corresponding codeword bit is referred to as the *a priori MI*, the MI between the *extrinsic* output and the corresponding codeword bit is referred to as the *extrinsic* MI, and the MI between the channel observation and the corresponding codeword bit is referred to as the *channel* MI.

***Algorithm** 2:* Convergence Check at Node $\boxed{\mathrm{G}_0}$

- Let I_A denote the *a priori* MI and I_E denote the *extrinsic* MI. Then the *a posteriori* MI $I_{\rm AP}$ is given by

$$I_{\rm AP} = J(\sqrt{[J^{-1}(I_A)]^2 + [J^{-1}(I_E)]^2}), \tag{9}$$

where the $J(\cdot)$ and $J^{-1}(\cdot)$ functions are given in [34], I_A is the *a priori* MI, and I_E is the *extrinsic* MI. As shown in Section III-C of [29], supposing that the *a posteriori* MI is Gaussian, an estimate of the BER p_{est} is then given by

$$p_{est} = {\rm Q}\left(J^{-1}(1 - I_{\rm AP})/2\right), \tag{10}$$

where

$${\rm Q}(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} \exp\left(-\frac{t^2}{2}\right) dt. \tag{11}$$

- If the estimated BER p_{est} is less than some preselected target BER, a local decoding success is declared; otherwise, a local decoding failure is declared.

For a fixed SNR E_b/N_0, the channel bit LLR corresponding to the binary-input AWGNC is Gaussian with variance [29]

$$\sigma^2_{\rm ch} = 8R_{\rm BMST}\frac{E_b}{N_0}, \tag{12}$$

where $R_{\rm BMST}$ is the rate of the BMST codes. The channel MI is then given by

$$I_{ch} = J(\sigma_{\rm ch}) = J\left(\sqrt{8R_{\rm BMST}\frac{E_b}{N_0}}\right). \tag{13}$$

The modified EXIT chart analysis algorithm of BMST codes, similar to the protograph-based EXIT chart analysis algorithm of SC-LDPC codes [31], can now be described as follows.

Algorithm *3:* EXIT Chart Analysis of BMST Codes with Window Decoding

- **Initialization:** All messages over those half-edges (connected to the channel) at nodes $\boxed{+}$ are initialized as I_{ch} according to (13), all messages over those half-edges (connected to the information source) at nodes $\boxed{\rm G_0}$ are initialized as 0, and all messages over the remaining (inter-connected) full-edges are initialized as 0. Set a maximum number of iterations $I_{\max}$.
- **Sliding window decoding:** For each window position, the $d+1$ decoding layers perform MI message processing/passing layer-by-layer according to the schedule

$$\boxed{+} \rightarrow \boxed{=} \rightarrow \boxed{\rm G_0} \rightarrow \boxed{=} \rightarrow \boxed{+}.$$

After a fixed number of iterations $I_{\max}$, perform a convergence check at node $\boxed{\rm G_0}$ using Algorithm 2. If a local decoding failure is declared, then window decoding terminates; otherwise, a local decoding success is declared, the window position is shifted, and decoding continues. A complete decoding success for a specific channel parameter E_b/N_0 and target BER is declared if and only if all target layers declare decoding successes.

Now we can denote the iterative decoding threshold $(E_b/N_0)^*$ of BMST code ensembles for a preselected target BER as the minimum value of the channel parameter E_b/N_0 which allows the decoder of Algorithm 3 to output a decoding success, in the limit of large code lengths (i.e., $B \to \infty$).

V. Impact of Parameters on BMST Codes

In this section we study the impact of various parameters (coupling width m, Cartesian product order B, and decoding delay d) on BMST codes. Three regimes are considered: (1) fixed m and B, increasing d, (2) fixed m and d, increasing B, and (3) fixed B, increasing m (and hence d).

All simulations are performed assuming BPSK modulation and an AWGNC. In the computation of the asymptotic window decoding thresholds of BMST codes, we set a maximum number of iterations $I_{\max} = 1000$. We will refer to the iterative decoding threshold $(E_b/N_0)^*$ simply as E_b/N_0 when it does not lead to ambiguity. In the simulations of finite-length performance, $m+1$ random interleavers (randomly generated but fixed) of size $n = NB$ are used for encoding. The iterative sliding window decoding algorithm [16, Algorithm 3] for BMST codes is performed using a layer-by-layer updating schedule with a maximum iteration number of 18, and the entropy stopping criterion [16, 35] with a preselected threshold of 10^{-6} is employed.

A. *Fixed m and B, Increasing d*

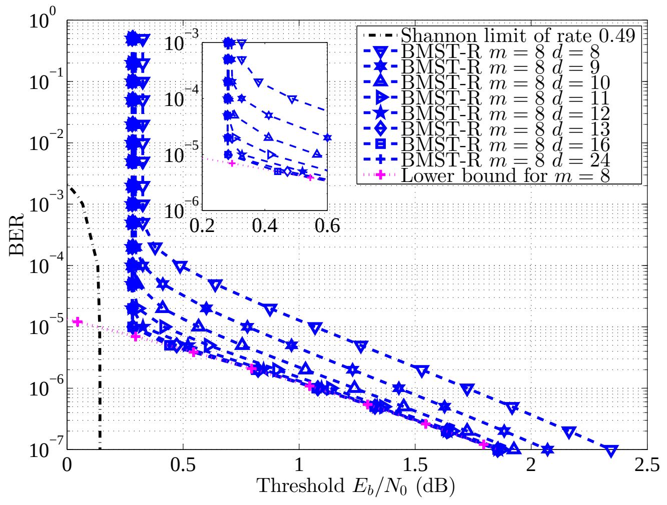
Example *1 (Asymptotic Performance):* Consider a $R_{\rm BMST} = 0.49$ BMST-R $[2,1]$ code ensemble with $m = 8$ and $L = 392$. We calculate its window decoding thresholds with different preselected target BERs and different decoding delays. The calculated thresholds in terms of the SNR E_b/N_0 versus the preselected target BERs together with the lower bound are shown in Fig. 5(a), where we observe that

1) In the waterfall region (above a critical BER), the thresholds remain almost constant. However, once the critical BER is reached, the thresholds increase as the target BER decreases.
2) For a small decoding delay (say $d = m$), the thresholds do not achieve the lower bound even in the high SNR region.
3) For a larger decoding delay (roughly $d = 2m \sim 3m$), the thresholds correspond to the lower bound in the high SNR region, suggesting that the window decoding algorithm with decoding delay $d \geq 2m \sim 3m$ is near optimal for BMST codes.
4) The error floor region threshold improves as the decoding delay d increases, but it does not improve much further beyond a certain decoding delay (roughly $d = 2m \sim 3m$).

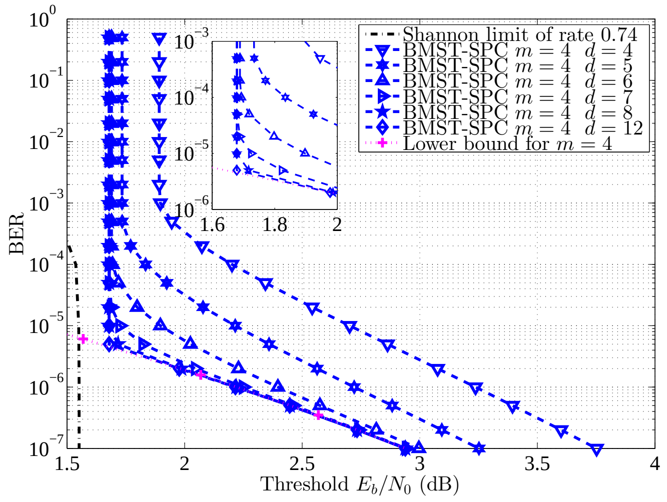
Similar behavior has also been observed for BMST-SPC code ensembles, as shown in Fig. 5(b), where the thresholds of a rate $R_{\rm BMST} = 0.74$ BMST-SPC $[4,3]$ code ensemble constructed with $m = 4$ and $L = 296$ and decoded with different decoding delays d are depicted.

The window decoding thresholds, corresponding to a preselected target BER[4] of 10^{-5}, for the $(3,6)$-regular SC-LDPC code ensemble with $m = 1$ and the BMST-R $[2,1]$ code

[4]We choose a BER of 10^{-5} for comparison because it represents a target BER commonly used in many practical applications.



(a)



(b)

Fig. 5. Window decoding thresholds in terms of E_b/N_0 (dB) with different target BERs and different decoding delays for (a) a rate $R_{\mathrm{BMST}} = 0.49$ BMST-R $[2, 1]$ code ensemble with $m = 8$ and $L = 392$, and (b) a rate $R_{\mathrm{BMST}} = 0.74$ BMST-SPC $[4, 3]$ code ensemble with $m = 4$ and $L = 296$.

ensemble with $m = 8$ as a function of decoding delay d is shown in Fig. 6. We see that, similar to the SC-LDPC code ensemble, the threshold of the BMST code ensemble improves as the decoding delay d increases and it becomes better than that of the SC-LDPC code ensemble beyond a certain decoding delay (roughly $d = 10$).

***Example** 2 (Finite-Length Performance):* Consider rate $R_{\mathrm{BMST}} = 0.49$ BMST-R $[2, 1]$ codes with $m = 8$ and $L = 392$. The BER performance of BMST-R codes decoded with different decoding delays d is shown in Fig. 7(a), where we observe that

1) The BER performance of BMST-R codes decoded with different delays d matches well with the corresponding window decoding thresholds in the high SNR region.
2) The BER performance in the waterfall region improves as the decoding delay d increases, but it does not improve much further beyond a certain decoding delay (roughly $d = 10$).

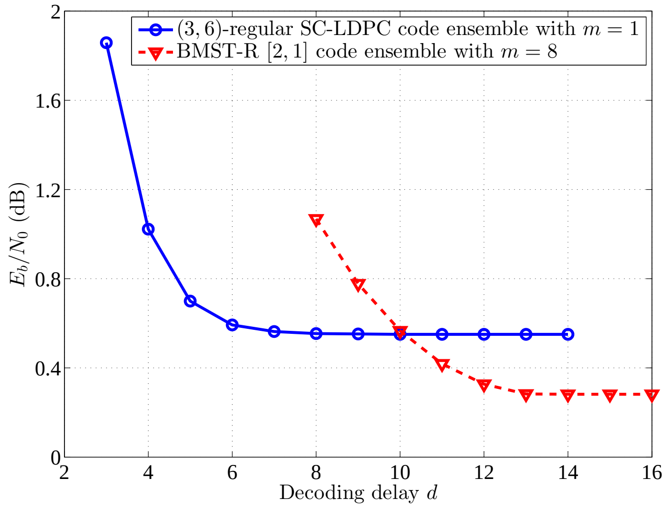


Fig. 6. Window decoding thresholds for a $(3, 6)$-regular SC-LDPC code ensemble with $m = 1$ and a BMST-R $[2, 1]$ code ensemble with $m = 8$ as a function of decoding delay d. The preselected target BER is 10^{-5}. The component submatrices for the SC-LDPC code ensemble are $\boldsymbol{B}_0 = [2\ 1]$ and $\boldsymbol{B}_1 = [1\ 2]$.

3) The error floor improves as the decoding delay d increases, and it matches well with the lower bound for BMST-R codes with $m = 8$ when d increases up to a certain point (roughly $d = 16$).

These results are consistent with the asymptotic threshold performance analysis shown in Fig. 5(a).

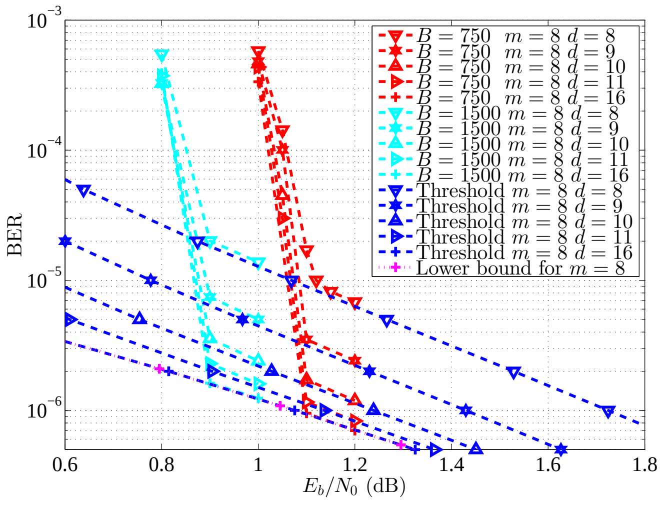
Similar behavior has also been observed for BMST-SPC code ensembles, as shown in Fig. 7(b), where the simulated decoding performance of a rate $R_{\mathrm{BMST}} = 0.74$ BMST-SPC $[4, 3]$ code constructed with $m = 4$, $L = 296$, and $B = 1200$, and decoded with different decoding delays d is depicted.

B. Fixed m and d, Increasing B

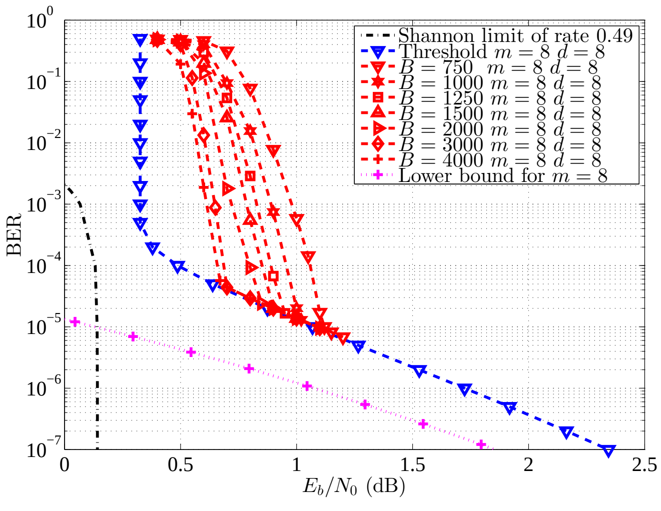
***Example** 3 (Finite-Length Performance):* Consider rate $R_{\mathrm{BMST}} = 0.49$ BMST-R $[2, 1]$ codes with $m = 8$ and $L = 392$. The BER performance of BMST-R codes constructed with different Cartesian product orders B is shown in Fig. 8, where we observe that

1) Similar to SC-LDPC codes, where increasing the lifting factor M improves waterfall region performance, increasing the Cartesian product order B of BMST codes also improves waterfall region performance. As expected, this improvement saturates for sufficiently large B. For example, the improvement at a BER of 10^{-5} from $B = 1000$ to $B = 2000$, both decoded with $d = 16$, is about 0.17 dB, while the improvement decreases to about 0.06 dB from $B = 3000$ to $B = 4000$.
2) The BER performance of BMST-R codes matches well with the corresponding window decoding thresholds in the error floor region.
3) The error floors, which are solely determined by the encoding memory m (see Section II-C), cannot be lowered by increasing B.

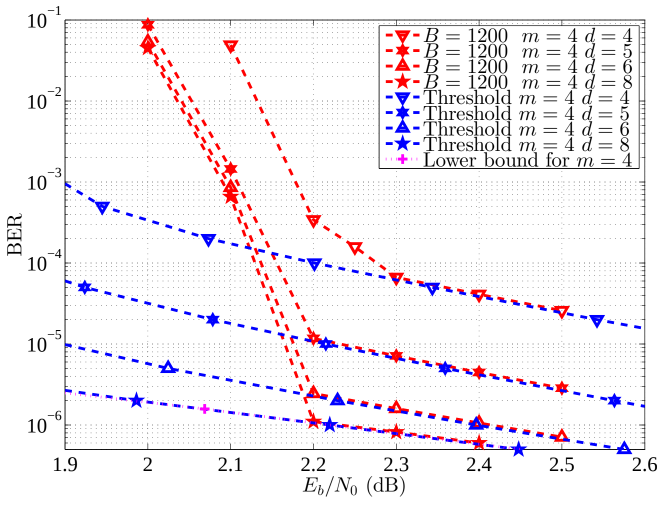
Remark: We found from simulations that, in the error floor region, the gap between finite-length performance and



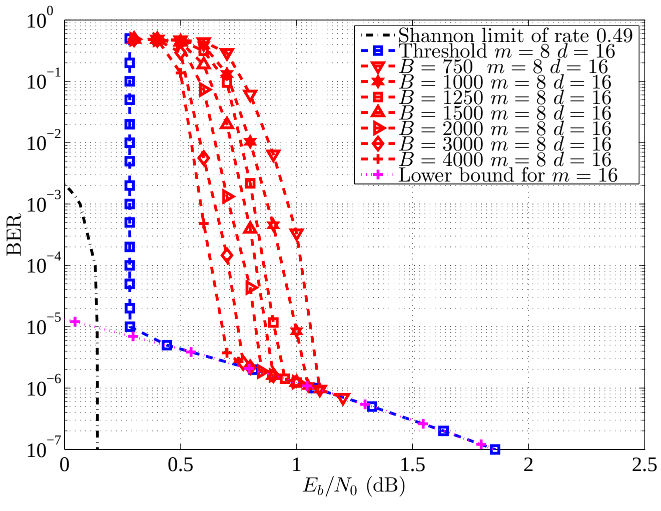
(a)



(a)



(b)



(b)

Fig. 7. Simulated decoding performance of BMST codes decoded with different decoding delays d. The corresponding window decoding thresholds and the lower bound are also plotted. (a) Rate $R_{\rm BMST} = 0.49$ BMST-R $[2, 1]$ codes with encoding memory $m = 8$ and coupling length $L = 392$. The Cartesian product orders of the two BMST-R codes are $B = 750$ and $B = 1500$, respectively. (b) A rate $R_{\rm BMST} = 0.74$ BMST-SPC $[4, 3]$ code with encoding memory $m = 4$, coupling length $L = 296$, and Cartesian product order $B = 1200$.

Fig. 8. Simulated decoding performance of rate $R_{\rm BMST} = 0.49$ BMST-R $[2, 1]$ codes with different Cartesian product orders B. The encoding memory $m = 8$ and the coupling length $L = 392$. The codes are decoded with (a) decoding delay $d = 8$, and (b) $d = 16$. The corresponding window decoding thresholds and the lower bound for BMST-R codes with $m = 8$ are also plotted.

window decoding threshold $(E_b/N_0)^*$ is less than 0.02 dB. For example, the values of E_b/N_0 needed to achieve a BER of 10^{-5} for a BMST-R $[2, 1]$ code with $m = 8$, very extremely large Cartesian product order (say, $B = 4000$), and decoding delay $d = 8$ is 1.087 dB, while the calculated window decoding threshold for a preselected target BER of 10^{-5} of the BMST-R $[2, 1]$ code ensemble with $m = 8$ and $d = 8$ is $(E_b/N_0)^* = 1.069$. This result again demonstrates that the finite-length performance is consistent with the asymptotic performance analysis.

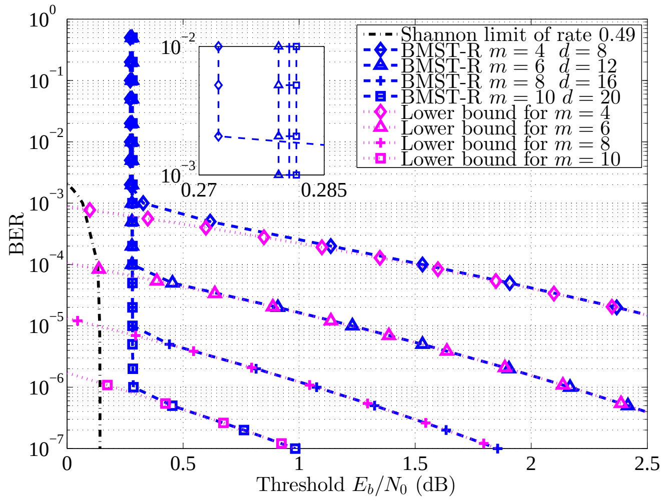
C. Fixed B, Increasing m (and hence d)

***Example** 4 (Asymptotic Performance):* Consider a family of $R_{\rm BMST} = 0.49$ BMST-R $[2, 1]$ code ensembles with different encoding memories m. The calculated window decoding thresholds in terms of the SNR E_b/N_0 versus the preselected target BERs together with the lower bounds are shown in Fig. 9(a), where we observe that

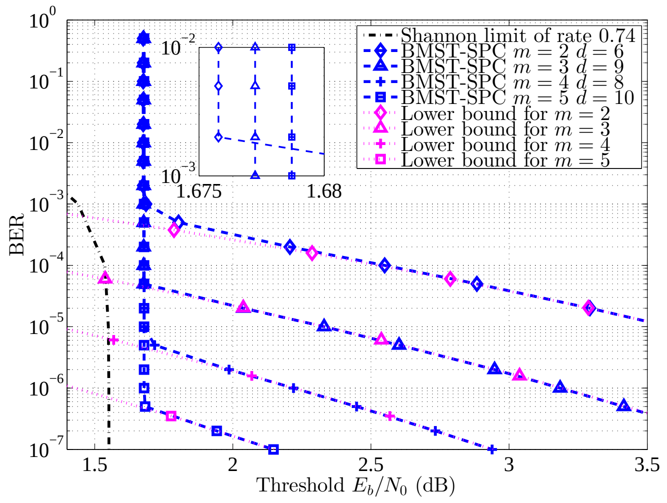
1) For a high target BER (roughly above 10^{-3}), the threshold with a sufficiently large decoding delay degrades slightly as the encoding memory m increases, due to errors propagating to successive decoding windows.
2) The error floor can be lowered by increasing the encoding memory m (and hence the decoding delay d).

Similar behavior has also been observed for BMST-SPC code ensembles, as shown in Fig. 9(b), where the thresholds of a family of rate $R_{\rm BMST} = 0.74$ BMST-SPC $[4, 3]$ code ensembles are depicted.

***Example** 5 (Finite-Length Performance):* Consider rate $R_{\rm BMST} = 0.49$ BMST-R $[2, 1]$ codes constructed with encoding memories $m = 4$, 6, 8, and 10, and Cartesian product orders $B = 750$ and $B = 1500$. The simulated BER



(a)



(b)

Fig. 9. Window decoding thresholds in terms of $(E_b/N_0)^*$ (dB) with different target BERs for two BMST code ensemble families with different encoding memories m. (a) BMST-R $[2, 1]$ code ensemble and (b) BMST-SPC $[4, 3]$ code ensemble.

performance with sufficiently large decoding delay is shown in Fig. 10, where we observe that

1) The BER performance in the waterfall region degrades slightly as the encoding memory m increases, due to errors propagating to successive decoding windows.
2) The error floor of the BER curves is lowered by increasing the encoding memory m (and hence the decoding delay d).

These results are consistent with the asymptotic performance analysis shown in Fig. 9(a).

VI. Performance and Complexity Comparison of SC-LDPC Codes and BMST Codes

In addition to decoding performance, the latency introduced by employing channel coding is a crucial factor in the design of a practical communication system. For example, minimizing latency is very important in applications such as personal wireless communication and real-time audio and video. In this

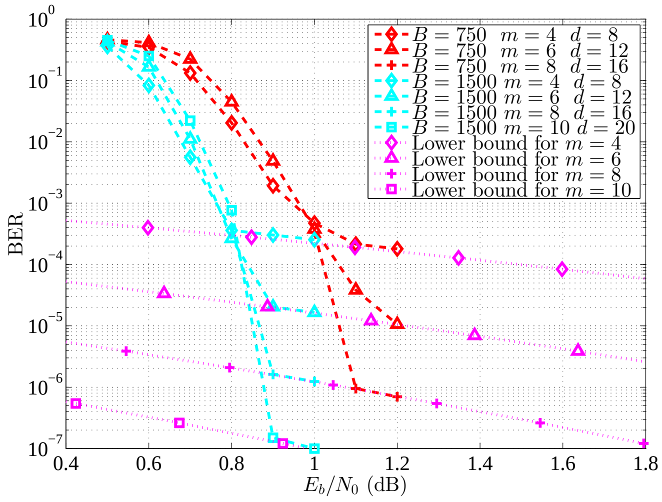


Fig. 10. Simulated decoding performance of rate $R_{\text{BMST}} = 0.49$ BMST-R $[2, 1]$ codes constructed with different encoding memories m and decoded with decoding delay $d = 2m$. The Cartesian product orders of the involved BMST-R codes are $B = 750$ and $B = 1500$. The corresponding lower bound for BMST-R codes is also plotted.

section, we first compare the performance of BMST codes and SC-LDPC codes when the two decoding latencies are equal. Then a computational complexity comparison is presented.

We restrict consideration to $(3, 6)$-regular SC-LDPC codes with coupling width $m = 1$, where two component submatrices $\mathbf{B}_0 = [2\ 1]$ and $\mathbf{B}_1 = [1\ 2]$ are used, due to their superior thresholds and finite-length performance with window decoding when the decoding delay is relatively small (see, e.g., [15, 31]). For the BMST codes, we consider BMST-R [2,1] codes with encoding memory $m = 8$, due to their near-capacity performance in the waterfall region and relatively low error floor (see Section V). In the simulations, the iterative sliding window decoding algorithm for SC-LDPC codes uses the uniform parallel (flooding) updating schedule with a maximum iteration number of 100, while for the BMST codes, window decoding is performed using the layer-by-layer updating schedule with a maximum iteration number of 18. The entropy stopping criterion [16, 35] is employed for both window decoding algorithms with a preselected threshold of 10^{-6}.

The decoding latency of the sliding window decoder, in terms of bits, is given by [15]

$$T_{\text{SC}} = 2M(d_{\text{SC}} + 1) \tag{14}$$

for the $(3, 6)$-regular SC-LDPC codes, and

$$T_{\text{BMST}} = 2B(d_{\text{BMST}} + 1) \tag{15}$$

for the BMST-R $[2, 1]$ codes, where d_{SC} and d_{BMST} are the decoding delays of the SC-LDPC codes and BMST codes, respectively. When the parameters M, B, d_{SC}, and d_{BMST} satisfy $B = M(d_{\text{SC}} + 1)/(d_{\text{BMST}} + 1)$, the decoding latency of BMST-R $[2, 1]$ codes is the same as that of $(3, 6)$-regular SC-LDPC codes. In our simulations, we consider decoding delay $d_{\text{SC}} = 5$ (i.e., window size $W = d_{\text{SC}} + 1 = 6$), which is a good choice for the SC-LDPC codes to achieve optimum

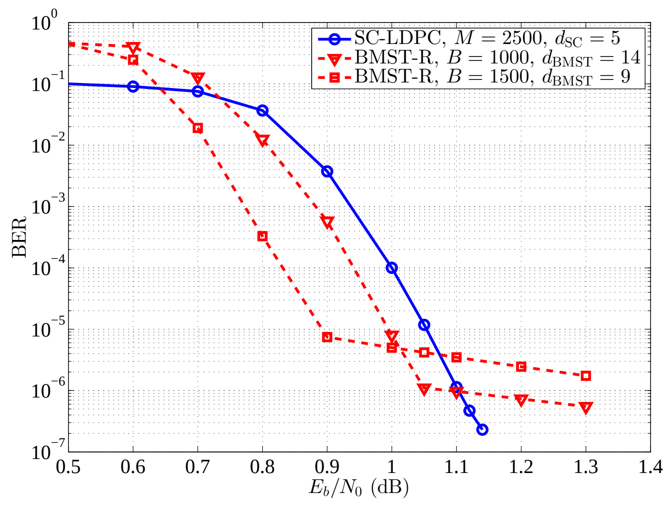


Fig. 11. Simulated decoding performance of BMST-R $[2, 1]$ codes with encoding memory $m = 8$ compared to $(3, 6)$-regular SC-LDPC code with coupling width $m = 1$. The values of B and d_{BMST} for the BMST-R codes are chosen in such a way that the decoding latencies of all the codes are the same.

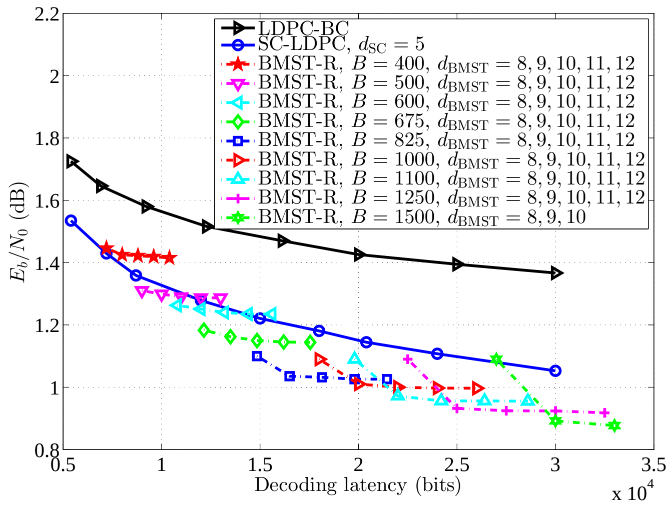


Fig. 12. Required E_b/N_0 to achieve a BER of 10^{-5} for $(3, 6)$-regular LDPC-BCs, $(3, 6)$-regular SC-LDPC codes, and BMST-R $[2, 1]$ codes as a function of decoding latency.

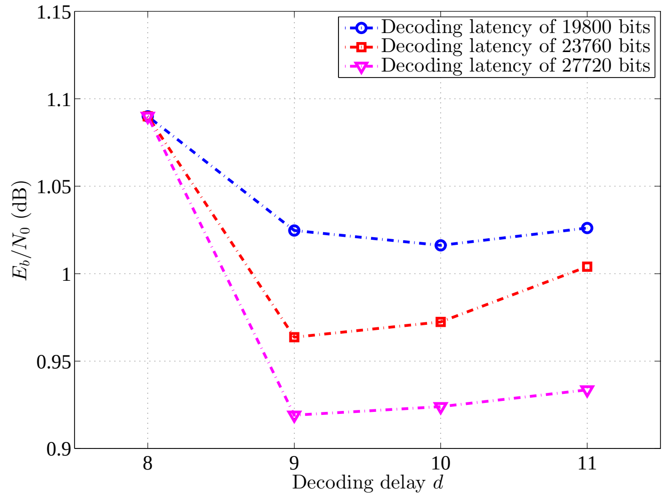


Fig. 13. Required E_b/N_0 to achieve a BER of 10^{-5} for BMST-R $[2, 1]$ codes with different decoding delays d_{BMST} and decoding latencies of 19800, 23760, and 27720 bits.

performance when the decoding latency is fixed [15].

A. Performance Comparison

In Fig. 11, BMST-R $[2, 1]$ codes are compared to $(3, 6)$-regular SC-LDPC codes, where the values of the Cartesian product order B and decoding delay d_{BMST} for the BMST-R codes are chosen such that the two decoding latencies T_{BMST} and T_{SC} are the same. We see that the BMST-R codes outperform the SC-LDPC code in the waterfall region but have a higher error floor. From Fig. 11, we also see that, in the waterfall region, the BMST-R code constructed with a larger Cartesian product order B and decoded with a smaller decoding delay $d_{\mathrm{BMST}} = 9$ outperforms the BMST-R code constructed with a smaller B and decoded with a larger decoding delay $d_{\mathrm{BMST}} = 14$ but has a higher error floor (both have the same decoding latency). In other words, selecting a smaller d_{BMST}, which is typically detrimental to decoder performance, is compensated for by allowing a larger B, which improves code performance. For example, at a BER of 10^{-5}, the BMST-R code with $B = 1000$ and decoded with decoding delay $d_{\mathrm{BMST}} = 14$ gains 0.05 dB compared to the equal latency SC-LDPC code with $M = 2500$, while the gain increases to 0.15 dB by using the BMST-R code with $B = 1500$ and $d_{\mathrm{BMST}} = 9$.

The E_b/N_0 required to achieve a BER of 10^{-5} for equal latency $(3, 6)$-regular LDPC-BCs, $(3, 6)$-regular SC-LDPC codes, and BMST-R $[2, 1]$ codes as a function of decoding latency is shown in Fig. 12, where we observe that both the BMST-R codes and the SC-LDPC codes perform significantly better than the LDPC-BCs. Also, the performance of the BMST-R codes (with fixed Cartesian product order B) improves as the decoding delay d_{BMST} (and hence the latency) increases, but it does not improve much further beyond a certain decoding delay (roughly $d_{\mathrm{BMST}} = 10$). (Note again that increasing the decoding delay d_{BMST} improves decoder performance and increasing the Cartesian product order B improves code performance.) However, under an equal decoding latency assumption, increasing the decoding delay d_{BMST} or the Cartesian product order B does not always lower the E_b/N_0 required to achieve a BER of 10^{-5}. For example, when the decoding latency is 14850 bits, the performance of the BMST-R code with $B = 825$ and decoded with $d_{\mathrm{BMST}} = 8$ is better than that of the BMST-R code with $B = 675$ and decoded with $d_{\mathrm{BMST}} = 10$. However, if we increase the latency to 19800 bits, the code with the Cartesian product order $B = 825$ and decoded with a larger $d_{\mathrm{BMST}} = 11$ still outperforms the code with $B = 1100$ and decoded with a smaller $d_{\mathrm{BMST}} = 8$. This raises the interesting question of how to choose B and d_{BMST} in order to achieve the best performance when the decoding latency of the sliding window decoder for BMST-R codes is fixed.

We also see from Fig. 12 that, for a fixed decoding latency roughly less than 15000 bits, to achieve a BER of 10^{-5},

$d_{\rm BMST} = 8$ is a good choice for optimum performance. This is due to the fact that the interleavers, which break short cycles in the normal graph of BMST codes, especially when the interleavers of size $n = NB$ are generated randomly, play a crucial role in iterative decoding [16]. That is, the larger the Cartesian product order B is, the better the performance of BMST codes becomes. However, the value of E_b/N_0 required to achieve a BER of 10^{-5} for BMST-R $[2,1]$ codes decoded with a fixed decoding delay $d_{\rm BMST}$ is bounded below by its corresponding window decoding threshold (see Section V-B).

Fig. 13 shows the E_b/N_0 values required for BMST-R [2,1] codes to achieve a BER of 10^{-5} with different decoding delays $d_{\rm BMST}$ and larger decoding latencies of 19800, 23760, and 27720 bits. Here we see that the required values of E_b/N_0 for the BMST-R $[2,1]$ codes with $d_{\rm BMST} = 8$ are the same and approach the corresponding window decoding threshold (as remarked in Section V-B). In this case, however, we also observe that the required values of E_b/N_0 continue to decreases until roughly $d_{\rm BMST} = 9 \sim 10$, and then they increase gradually as the decoding delay $d_{\rm BMST}$ increases further. This increase results from the fact that the improved decoder performance obtained by increasing $d_{\rm BMST}$ is not compensating for the decrease in code performance as a result of the smaller Cartesian product order B. Thus, for larger decoding latencies (up to 35000 bits), $d_{\rm BMST} = 9$ is a good choice for optimum performance.

B. Complexity Comparison

As shown in [16], we can measure the computational complexity of BMST codes by the total number of operations. Consider a BMST-R $[N,1]$ code or a BMST-SPC $[N,N-1]$ code with Cartesian product order B and decoding delay $d_{\rm BMST}$. Let $Opt(\mathcal{A})$ denote the number of operations at a generic node $\mathcal{A}$. Each decoding layer has NB parallel nodes $\boxed{=}$, NB parallel nodes $\boxed{+}$, and a node of type $\boxed{\rm G}$. The computational complexity for each node $\boxed{=}$, each node $\boxed{+}$, and each node $\boxed{\rm G}$ is $\mathcal{O}(m+2)$, $\mathcal{O}(m+1)$, and $\mathcal{O}(NB)$, respectively. Thus, the total number of operations for each decoding layer update is given by

$$\begin{aligned} & NB \cdot Opt\left(\boxed{=}\right) + NB \cdot Opt\left(\boxed{+}\right) + Opt\left(\boxed{\rm G}\right) \\ & = NB(m+2) + NB(m+1) + NB = NB(2m+4). \end{aligned} \tag{16}$$

Let $I_{\rm BMST}$ denote the average number of iterations required to decode a target layer for BMST codes. Since each iteration requires both a forward recursion ($d_{\rm BMST}$ layer-updates) and a backward recursion ($d_{\rm BMST}$ layer-updates), the total (average) computational complexity per window is given by

$$\begin{aligned} & \mathcal{O}(NB(2m+4) \times 2d_{\rm BMST}) I_{\rm BMST} \\ & = \mathcal{O}(NB(4m+8)d_{\rm BMST}) I_{\rm BMST}. \end{aligned} \tag{17}$$

Note that the number of decoded (target) bits for the window decoder at each time instant is NB, and thus the computational complexity per decoded bit for a BMST code is

$$\begin{aligned} & \mathcal{O}(NB(4m+8)d_{\rm BMST}) I_{\rm BMST}/(NB) \\ & = \mathcal{O}\left((4m+8) d_{\rm BMST} I_{\rm BMST}\right). \end{aligned} \tag{18}$$

Now consider a $(3,6)$-regular SC-LDPC code with lifting factor M and decoding delay $d_{\rm SC}$ (the corresponding decoding window size $W = d_{\rm SC}+1$). Let $I_{\rm SC}$ denote the average number of iterations required to decode a target layer for SC-LDPC codes. Note that the numbers of operations at a variable node and a check node of $(3,6)$-regular SC-LDPC codes are 3 and 6, respectively. The average computational complexity (also measured by the total number of operations) per window is then given by

$$\mathcal{O}\left(3T_{\rm SC} + 6T_{\rm SC}/2\right) I_{\rm SC} = \mathcal{O}\left(6 T_{\rm SC} I_{\rm SC}\right), \tag{19}$$

where $T_{\rm SC}$ is the decoding latency. Note that the number of decoded (target) bits for the window decoder at each time instant is $T_{\rm SC}/(d_{\rm SC}+1)$, and thus the computational complexity per decoded bit for a $(3,6)$-regular SC-LDPC code is

$$\frac{\mathcal{O}\left(6T_{\rm SC}\right) I_{\rm SC}}{T_{\rm SC}/(d_{\rm SC}+1)} = \mathcal{O}\left(6(d_{\rm SC}+1) I_{\rm SC}\right). \tag{20}$$

Table II shows the average computational complexity per decoded bit of the $(3,6)$-regular SC-LDPC code and the BMST-R $[2,1]$ codes used in Fig. 11 that achieve a BER of 10^{-5} with a decoding latency of 30000 bits. The simulation parameters M, B, m, $d_{\rm BMST}$, $d_{\rm SC}$, $I_{\rm BMST}$ and $I_{\rm SC}$ are also included. We observe that, though the average number of iterations $I_{\rm BMST}$ for the BMST code is significantly less than $I_{\rm SC}$ for SC-LDPC code, the computational complexity per decoded bit for the BMST codes is higher than for the SC-LDPC code. However, the BMST codes outperform the SC-LDPC code in the waterfall region (see Fig. 11 in Section VI-B). This means that BMST-R $[2,1]$ codes, compared to $(3,6)$-regular SC-LDPC codes, obtain performance gains at a cost of higher computational complexity.

VII. Conclusions

In this paper, we described BMST codes using both an algebraic description and a graphical representation for the purpose of showing that BMST codes can be viewed as a class of SC codes. Then, based on a modified EXIT chart analysis and finite-length computer simulations, we investigated the impact of several parameters (coupling width, Cartesian product order, and decoding delay) on the performance of BMST codes. We then examined the relationship between the Cartesian product order, the decoding delay, and the decoding performance of BMST codes for fixed decoding latency in comparison to SC-LDPC codes, and a comparison of computational complexity was also presented. It was observed that, under the equal decoding latency constraint, BMST codes using the repetition $[2,1]$ code (BMST-R $[2,1]$ code) as the basic code can outperform $(3,6)$-regular SC-LDPC codes in the waterfall region but have a higher error floor and a larger decoding complexity. An interesting future research topic to complement the work reported here is to embed a partial superposition strategy into the code design to further improve the performance of the original BMST codes for a given decoding latency.

Acknowledgment

The authors would like to thank Prof. Daniel J. Costello, Jr. for his helpful comments, polishing this paper, and invaluable

TABLE II
COMPUTATIONAL COMPLEXITY PER DECODED BIT OF A $(3,6)$-REGULAR SC-LDPC CODE AND BMST-R $[2,1]$ CODES THAT ACHIEVE A BER OF 10^{-5} WITH DECODING LATENCY OF 30000 BITS

Codes	$M\backslash B$	m	$d_{\mathrm{SC}}\backslash d_{\mathrm{BMST}}$	$I_{\mathrm{SC}}\backslash I_{\mathrm{BMST}}$	Complexity
SC-LDPC	2500	1	5	9.65	347.4
BMST	1000	8	14	2.03	1136.8
BMST	1500	8	9	3.20	1152.0

contributions as a co-author of the conference version of this paper [36]. They would also like to thank Mr. Chulong Liang from Sun Yat-sen University for helpful discussions.

REFERENCES

[1] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.

[2] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[3] M. Lentmaier, A. Sridharan, D. J. Costello, Jr., and K. S. Zigangirov, "Iterative decoding threshold analysis for LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 5274–5289, Oct. 2010.

[4] S. Kudekar, T. J. Richardson, and R. L. Urbanke, "Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC," *IEEE Trans. Inf. Theory*, vol. 57, no. 4, pp. 803–834, Feb. 2011.

[5] ——, "Spatially coupled ensembles universally achieve capacity under belief propagation," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7761–7813, Dec. 2013.

[6] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, Jr., "Spatially coupled LDPC codes constructed from protographs," 2014, submitted to *IEEE Trans. Inf. Theory*. [Online]. Available: http://arxiv.org/abs/1407.5366

[7] A. E. Pusane, R. Smarandache, P. O. Vontobel, and D. J. Costello, Jr., "Deriving good LDPC convolutional codes from LDPC block codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 835–857, Feb. 2011.

[8] M. Lentmaier, M. M. Prenda, and G. P. Fettweis, "Efficient message passing scheduling for terminated LDPC convolutional codes," in *Proc. IEEE Int. Symp. on Inf. Theory*, St. Petersburg, Russia, Aug. 2011, pp. 1826–1830.

[9] N. ul Hassan, A. E. Pusane, M. Lentmaier, G. P. Fettweis, and D. J. Costello, Jr., "Reduced complexity window decoding schedules for coupled LDPC codes," in *Proc. IEEE Inf. Theory Workshop*, Lausanne, Switzerland, Sept. 2012, pp. 20–24.

[10] I. Andriyanova and A. Graell i Amat, "Threshold saturation for nonbinary SC-LDPC codes on the binary erasure channel," 2013, submitted to *IEEE Trans. Inf. Theory*. [Online]. Available: http://arxiv.org/abs/1311.2003

[11] D. G. M. Mitchell, A. E. Pusane, and D. J. Costello, Jr., "Minimum distance and trapping set analysis of protograph-based LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 1, pp. 254–281, Jan. 2013.

[12] D. G. M. Mitchell, M. Lentmaier, A. E. Pusane, and D. J. Costello, Jr., "Randomly punctured spatially coupled LDPC codes," in *Proc. Int. Symp. Turbo Codes Iterative Inf. Process.*, Aug. 2014, pp. 1–6.

[13] D. J. Costello, Jr., L. Dolecek, T. E. Fuja, J. Kliewer, D. G. M. Mitchell, and R. Smarandache, "Spatially coupled sparse codes on graphs: Theory and practice," *IEEE Communications Magazine*, vol. 52, no. 7, pp. 168–176, July 2014.

[14] P. M. Olmos and R. L. Urbanke, "A scaling law to predict the finite-length performance of spatially coupled LDPC codes," 2014, submitted to *IEEE Trans. Inf. Theory*. [Online]. Available: http://arxiv.org/abs/1404.5719

[15] K. Huang, D. G. M. Mitchell, L. Wei, X. Ma, and D. J. Costello, Jr., "Performance comparison of LDPC block and spatially coupled codes over GF(q)," *IEEE Trans. Commun.*, vol. 63, no. 3, pp. 592–604, Mar. 2015.

[16] X. Ma, C. Liang, K. Huang, and Q. Zhuang, "Block Markov superposition transmission: Construction of big convolutional codes from short codes," *IEEE Trans. Inf. Theory*, 2015, to appear.

[17] C. Liang, X. Ma, Q. Zhuang, and B. Bai, "Spatial coupling of generator matrices: A general approach to design good codes at a target BER," *IEEE Trans. Commun.*, vol. 62, no. 12, pp. 4211–4219, Dec. 2014.

[18] A. J. Feltström, D. Truhachev, M. Lentmaier, and K. S. Zigangirov, "Braided block codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2640–2658, June 2009.

[19] W. Zhang, M. Lentmaier, K. S. Zigangirov, and D. J. Costello, Jr., "Braided convolutional codes: A new class of turbo-like codes," *IEEE Trans. Inf. Theory*, vol. 56, pp. 316–331, Jan. 2010.

[20] S. Moloudi and M. Lentmaier, "Density evolution analysis of braided convolutional codes on the erasure channel," in *Proc. IEEE Int. Symp. on Inf. Theory*, Honolulu, HI, June 2014, pp. 2609–2613.

[21] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, "Staircase codes: FEC for 100 Gb/s OTN," *J. Lightwave Technol.*, vol. 30, no. 1, pp. 110–117, Jan. 2012.

[22] S. Moloudi, M. Lentmaier, and A. Graell i Amat, "Spatially coupled turbo codes," in *Proc. Int. Symp. Turbo Codes Iterative Inf. Process.*, Bremen, Germany, Aug. 2014, pp. 82–86.

[23] D. Divsalar, H. Jin, and R. J. McEliece, "Coding theorems for 'turbo-like' codes," in *Proc. Allerton Conf.*, Urbana, IL, Sept. 1998, pp. 201–210.

[24] H. D. Pfister and P. H. Siegel, "The serial concatenation of rate-1 codes through uniform random interleavers," *IEEE Trans. Inf. Theory*, vol. 49, no. 6, pp. 1425–1438, June 2003.

[25] A. Abbasfar, D. Divsalar, and K. Yao, "Accumulate-repeat-accumulate codes," *IEEE Trans. Commun.*, vol. 55, no. 4, pp. 692–702, Apr. 2007.

[26] A. R. Iyengar, M. Papaleo, P. H. Siegel, J. K. Wolf, A. Vanelli-Coralli, and G. E. Corazza, "Windowed decoding of protograph-based LDPC convolutional codes over erasure channels," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2303–2320, Apr. 2012.

[27] C. Liang, J. Hu, X. Ma, and B. Bai, "A new class of multiple-rate codes based on block Markov superposition transmission," 2014, submitted to *IEEE Trans. Signal Process.* [Online]. Available: http://arxiv.org/abs/1406.2785

[28] J. Hu, X. Ma, and C. Liang, "Block Markov superposition transmission of repetition and single-parity-check codes," *IEEE Commun. Lett.*, vol. 19, no. 2, pp. 131–134, Feb. 2015.

[29] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[30] G. Liva and M. Chiani, "Protograph LDPC codes design based on EXIT analysis," in *Proc. IEEE Global Commun. Conf.*, Washington, DC, Nov. 2007, pp. 3250–3254.

[31] L. Wei, D. G. M. Mitchell, T. E. Fuja, and D. J. Costello, Jr., "Design of spatially coupled LDPC codes over GF(q) for windowed decoding," 2014, submitted to *IEEE Trans. Inf. Theory*. [Online]. Available: http://arxiv.org/abs/1411.4373

[32] A. E. Pusane, A. J. Felström, A. Sridharan, M. Lentmaier, K. S. Zigangirov, and D. J. Costello, Jr., "Implementation aspects of LDPC convolutional codes," *IEEE Trans. Commun.*, vol. 56, no. 7, pp. 1060–1069, July 2008.

[33] G. D. Forney, Jr., "Codes on graphs: Normal realizations," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 520–548, Feb. 2001.

[34] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, no. 4, pp. 670–678, Apr. 2004.

[35] X. Ma and L. Ping, "Coded modulation using superimposed binary codes," *IEEE Trans. Inf. Theory*, vol. 50, pp. 3331–3343, Dec. 2004.

[36] K. Huang, X. Ma, and D. J. Costello, Jr., "EXIT chart analysis of block Markov superposition transmission of short codes," 2015, accepted by *Proc. IEEE Int. Symp. on Inf. Theory*. [Online]. Available: http://arxiv.org/abs/1502.00079